RZ/T1 グループ
マルチポート対応 ETHERNET ドライバ

R01AN3501JJ0110
Rev.1.10
2017.05.16

要旨

本アプリケーションノートでは、RZ/T1 グループ MCU を搭載した RSK RZ/T1 評価ボード対応の ETHERNET 機能サンプルプログラムについて説明します。

評価ボードに接続されたデバイスから ping コマンドを発行することで、評価ボードからの応答を確認することができます。

対象デバイス

RZ/T1

目次

1. 仕様

「表 1-1　使用する周辺機能と用途」、「**エラー! 参照元が見つかりません。**」を以下に示します。

表 1-1　使用する周辺機能と用途

周辺機能	用途
RZ/T1 内蔵 Cortex-R4F	プログラム動作用
RZ/T1 内蔵密結合メモリ(TCM)	Cortex-R4F プログラム・コード実行、プログラム・データ格納領域
RZ/T1 イーサネット MAC(ETHERC)	Ethernet ポートを使用してデータを送受信するために使用します。
RZ/T1 イーサネットスイッチ (ETHSW)	イーサネットスイッチ機能を使用します。

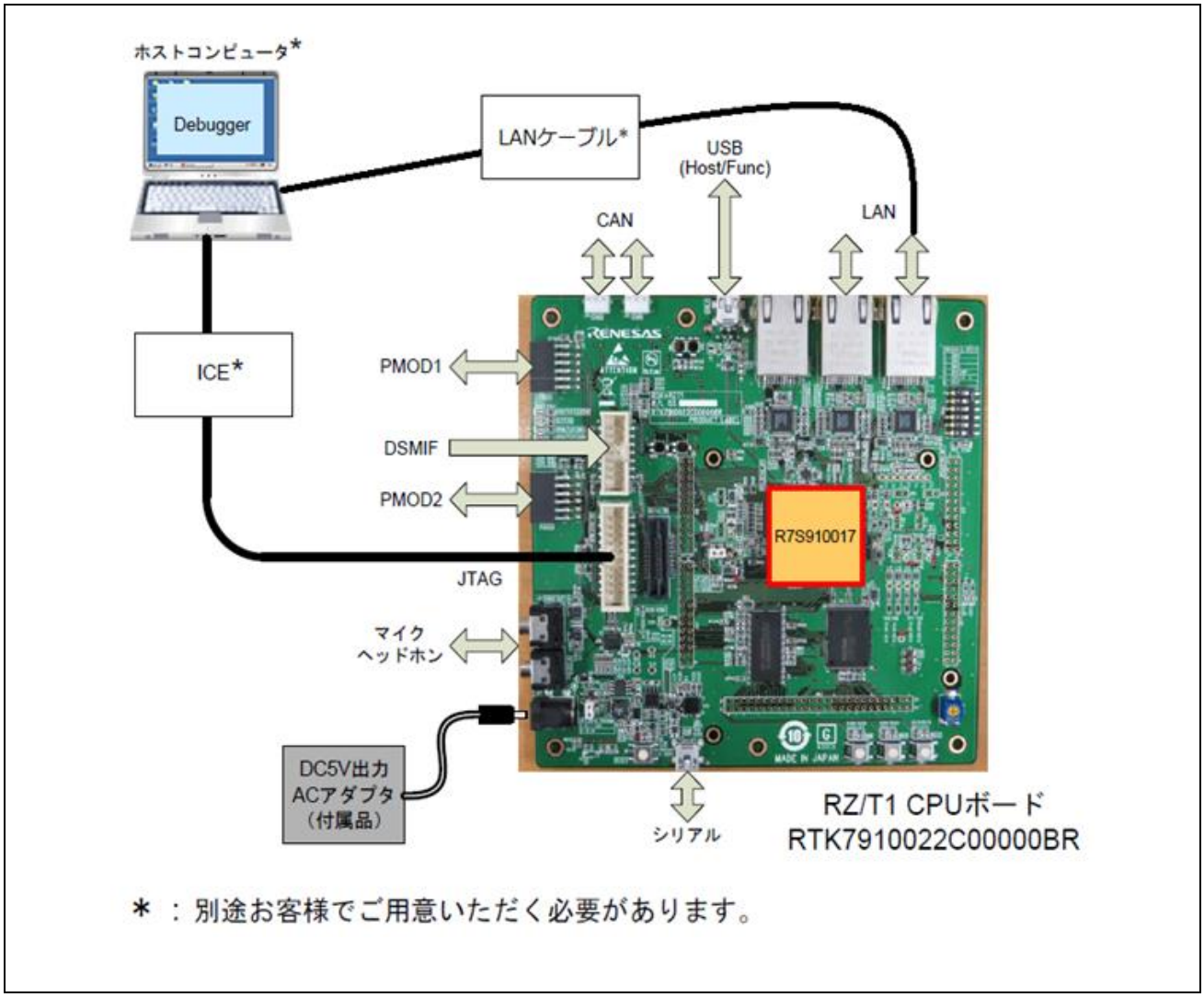


図 1.1　　システム構成例

2. 動作環境

本書のドライバは、下記の動作環境を想定しています。

表 2-1　動作環境

項目	内容
使用マイコン	RZ/T1 R7S910017
動作周波数	CPU クロック（Cortex-R4F）：450MHz
動作電圧	電源電圧（I/O）:3.3V 電源電圧（内部）:1.2V
統合開発環境	IAR システムズ製 Embedded Workbench for ARM V7.60.1
エミュレータ	IAR ARM 用 JTAG エミュレータ I-jet
動作モード	SPI ブートモード 16 ビットバスブートモード
使用ボード	RZ/T1 Evaluation Board (RTK7910022C00000BR)

3. 関連ドキュメント

本書に関連する文書を以下に示します。併せて参照してください。

- RZ/T1 グループ ユーザーズマニュアル ハードウェア編（R01UH0483JJ）
- RZ/T1 グループ 初期設定 アプリケーションノート（R01AN2554JJ）

4. ハードウェア説明

4.1 使用端子一覧

表 4-1　使用端子と機能

端子名	入出力	機能
ETH0_TXC ETH1_TXC	入力	10M／100M 送信クロック（2.5MHz／25MHz）入力端子
ETH0_TXEN ETH1_TXEN	出力	送信イネーブル信号出力端子
ETH0_TXER ETH1_TXER	出力	送信エラー信号出力端子
ETH0_TXD0～3 ETH1_TXD0～3	出力	送信データ信号出力端子
ETH0_RXC ETH1_RXC	入出力	受信クロック入出力端子
ETH0_RXDV ETH1_RXDV	入力	受信データイネーブル信号入力端子
ETH0_RXER ETH1_RXER	入力	受信データエラー信号入力端子
ETH0_RXD0～3 ETH1_RXD0～3	入力	受信データ信号入力端子
ETH0_CRS ETH1_CRS	入力	キャリアセンス信号入力端子
ETH0_COL ETH1_COL	入力	衝突検出信号入力端子
ETH_MDC	出力	マネージメントインタフェースクロック出力端子
ETH_MDIO	入出力	マネージメントデータ信号入出力端子
PHYLINK0 PHYLINK1	入力	PHY Link 信号入力端子
ETHSWSECOUT	出力	Ether Switch の 1 秒毎のイベント出力端子
PHYRESETOUT#	出力	PHY RESET 用出力信号（Ether0, Ether1 用）

4.2　参考回路

「図 4-1　ETHERNET 関連ブロック図」を以下に示します。

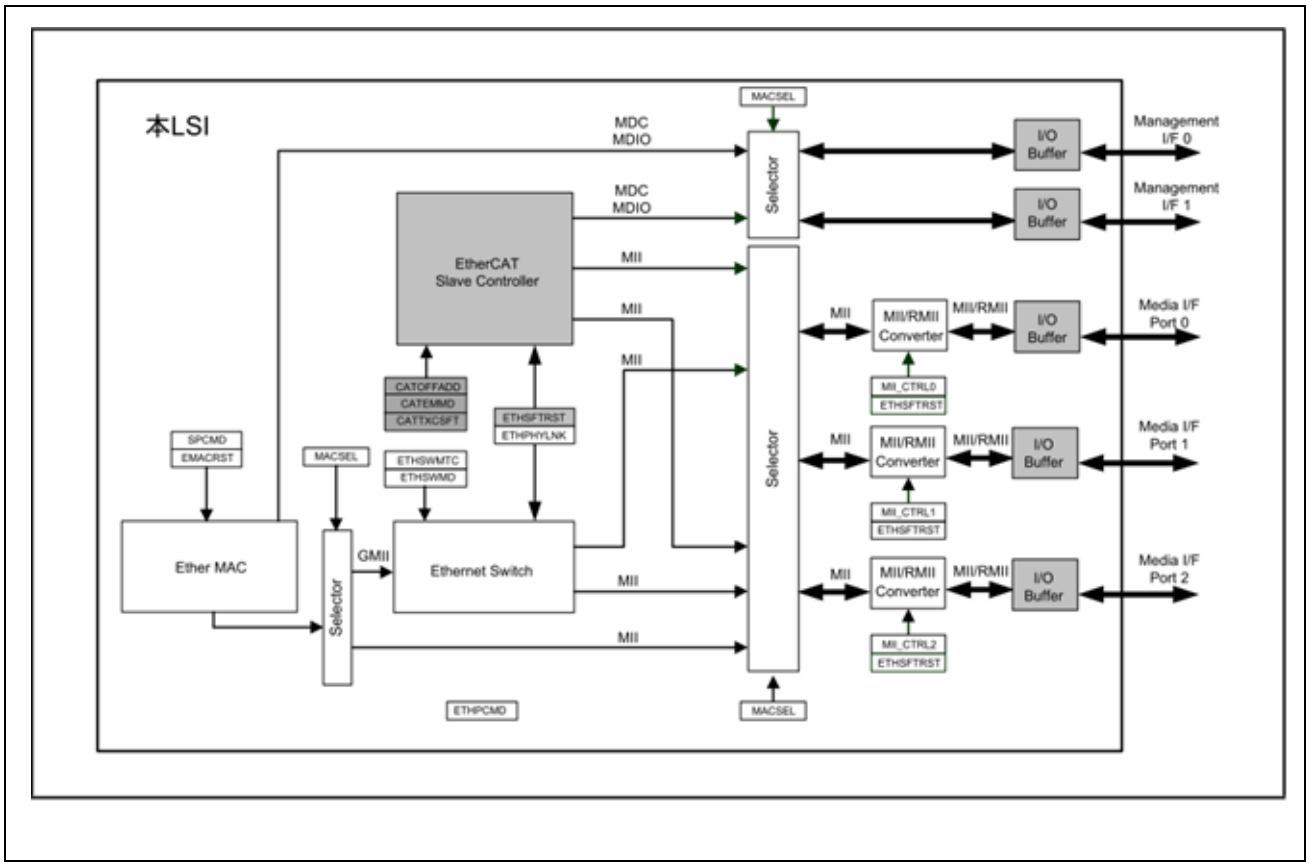


図 4-1　ETHERNET 関連ブロック図

5. ソフトウェア説明(ドライバ)

5.1 ドライバ概要

本サンプルプログラムでは、イーサネット MAC（ETHERC）、イーサネット PHY(ETHPHY)、イーサネットスイッチ（ETHSW）機能の Ethernet 関連モジュールを用いた、Ethernet 通信制御を行うためのドライバを提供します。

5.2 ファイル一覧

サンプルプログラムで使用しているドライバを以下に示します。

表 5-1　サンプルプログラムで使用するファイル

ファイル名	概要	備考
.. *1/inc/eth_hwfnc.h	ハードウェアファンクション定義ヘッダ	
.. *1/inc/r_eth.h	Ethernet 関連ドライバ公開ヘッダ	
.. *1/inc/r_eth_mac.h	EthernetMAC ドライバ定義ヘッダ	
.. *1/inc/r_eth_phy.h	EthernetPHY ドライバ定義ヘッダ	
.. *1/inc/r_eth_sw.h	EthernetSwitch ドライバ定義ヘッダ	
.. *1/src/drv/eth/eth_hwfnc.c	ハードウェアファンクションドライバ API 実装	
.. *1/src/drv/eth/r_eth_mac.c	EthernetMAC ドライバ API 実装	
.. *1/src/drv/eth/r_eth_phy.c	EthernetPHY ドライバ API 実装	
.. *1/src/drv/eth/r_eth_sw.c	EthernetSwitch ドライバ API 実装	

【注】　..*1　使用するサンプルプログラムのフォルダ名です。

5.3 基本データ型

サンプルプログラムで使用している基本データ型を表 5-2 に示します。

表 5-2　基本データ型

シンボル	内容
int8_t	8 ビット整数、符号あり（標準ライブラリにて定義）
int16_t	16 ビット整数、符号あり（標準ライブラリにて定義）
int32_t	32 ビット整数、符号あり（標準ライブラリにて定義）
int64_t	64 ビット整数、符号あり（標準ライブラリにて定義）
uint8_t	8 ビット整数、符号なし（標準ライブラリにて定義）
uint16_t	16 ビット整数、符号なし（標準ライブラリにて定義）
uint32_t	32 ビット整数、符号なし（標準ライブラリにて定義）
uint64_t	64 ビット整数、符号なし（標準ライブラリにて定義）

5.4　定数一覧

表 5-3　定数一覧

定義名	設定値	内容	定義ファイル
USE_ETHSW	1	イーサネットスイッチ機能オプション	r_eth.h
USE_ETHSW_MGTAG	1	イーサネットスイッチ　マネジメントタグ機能オプション	r_eth.h
MULTICAST_MODE_ENA	1	マルチキャストフレーム受信オプション	r_eth.h
PROMISCUOUS_MODE_ENA	1	全てのフレーム受信オプション	r_eth.h
MAC0_TYPE_MASK	0x01	MAC アドレスタイプのマスク値	r_eth.h
MAC0_TYPE_UCAST	0x00	ユニキャスト MAC アドレス	r_eth.h
MAC0_TYPE_MCAST	0x01	マルチキャスト MAC アドレス	r_eth.h
ETH_HEADR_SIZE	14	Ethernet フレームヘッダサイズ	r_eth.h
ETH_VLAN_HEADR_SIZE	18	VLAN タグつき Ethernet フレームヘッダサイズ	r_eth.h
ETH_EVT_PHYINTn	0x00000001 0x00000002 0x00000004	Ether PHYn イベント (n=0 – 2)	r_eth.h
ETH_EVT_RXDMACMP	0x00000100	Ether MACDMA 受信完了	r_eth.h
ETH_EVT_RXDMAERR	0x00000200	Ether MACDMA 受信エラー	r_eth.h
ETH_EVT_RXFIFOOVF	0x00000400	Ether RX-FIFO オーバーフロー	r_eth.h
ETH_EVT_TXDMACMP	0x00010000	Ether MACDMA 送信完了	r_eth.h
ETH_EVT_TXDMAERR	0x00020000	Ether MACDMA 送信エラー	r_eth.h
ETH_EVT_TXCMP	0x00100000	Ether 送信完了	r_eth.h
ETH_EVT_TXFIFOUDF	0x00200000	Ether TX-FIFO アンダーフロー	r_eth.h
ETH_EVT_TXFIFOERR	0x00400000	Ether TX-FOFO エラー	r_eth.h
ETH_EVT_PHY	ETH_EVT_PHYINT0 | ETH_EVT_PHYINT1 | ETH_EVT_PHYINT2	Ether PHY イベント	r_eth.h
ETH_EVT_TX	ETH_EVT_TXDMACMP | ETH_EVT_TXCMP | ETH_EVT_TXDMAERR | ETH_EVT_TXFIFOUDF | ETH_EVT_TXFIFOERR	Ether 送信関連イベント	r_eth.h
ETH_TYPE_IPv4	0x0800	Ethernet Type　: IPv4	r_eth.h
ETH_TYPE_ARP	0x0806	Ethernet Type　: ARP	r_eth.h
ETH_PORT0	1	送受信ポート指定(Ethernet ポート 0)	r_eth_mac.h
ETH_PORT1	2	送受信ポート指定(Ethernet ポート 1)	r_eth_mac.h
ETH_PORTALL	(ETH_PORT0 | ETH_PORT1)	送受信全ポート指定	r_eth_mac.h
MACINFO_TBLEND	0xFF	eth_macinfo_t 構造体の終端設定値	r_eth_mac.h
MACINFO_UCAST_MAX	2	eth_macinfo_t 構造体のユニキャスト MAC アドレス登録上限	r_eth_mac.h
htonl	ntohl	ホストバイトオーダーをネットワークバイトオーダーに変換(32bit)	r_eth_mac.h
htons	ntohs	ホストバイトオーダーをネットワークバイトオーダーに変換(16bit)	r_eth_mac.h

定義名	設定値	内容	定義ファイル
PHY_ADR0	1	PHY レジスタアクセス用 PHY0 アドレス	r_eth_phy.h
PHY_ADR1	2	PHY レジスタアクセス用 PHY1 アドレス	r_eth_phy.h
PHY_MAX_NUM	2	PHY インターフェース数	r_eth_phy.h
PHY_LINK_MASK	0x80004000	PHY LINK 状態マスク値	r_eth_phy.h
PHY_LINK_UP	0x00004000	PHY LINK-UP	r_eth_phy.h
PHY_LINK_DOWN	0x00000000	PHY LINK-DOWN	r_eth_phy.h
PHY_AUTONEGO_MASK	0x80002000	PHY Auto-Negotiation 状態マスク値	r_eth_phy.h
PHY_AUTONEGO_EN	0x00002000	PHY Auto-Negotiation 有効	r_eth_phy.h
PHY_AUTONEGO_DS	0x00000000	PHY Auto-Negotiation 無効	r_eth_phy.h
PHY_SPEED_MASK	0x80000030	PHY 速度状態マスク値	r_eth_phy.h
PHY_SPEED_10M	0x00000000	10BASE-T	r_eth_phy.h
PHY_SPEED_100M	0x00000010	100BASE-T	r_eth_phy.h
PHY_SPEED_1G	0x00000020	1000BASE-T	r_eth_phy.h
PHY_DUPLEX_MASK	0x80000001	PHY Duplex タイプマスク値	r_eth_phy.h
PHY_DUPLEX_HALF	0x00000000	PHY 半二重	r_eth_phy.h
PHY_DUPLEX_FULL	0x00000001	PHY 全二重	r_eth_phy.h
LAN_MODE_MSK	(PHY_SPEED_MASK | PHY_DUPLEX_MASK)	PHY 設定マスク値	r_eth_phy.h
LAN_10T_HD	(PHY_SPEED_10M | PHY_DUPLEX_HALF)	10BASE-T 半二重	r_eth_phy.h
LAN_10T_FD	(PHY_SPEED_10M | PHY_DUPLEX_FULL)	10BASE-T 全二重	r_eth_phy.h
LAN_100TX_HD	(PHY_SPEED_100M | PHY_DUPLEX_HALF)	100BASE-TX 半二重	r_eth_phy.h
LAN_100TX_FD	(PHY_SPEED_100M | PHY_DUPLEX_FULL)	100BASE-TX 全二重	r_eth_phy.h
LAN_1000T_HD	(PHY_SPEED_1G | PHY_DUPLEX_HALF)	1000BASE-T 半二重	r_eth_phy.h
LAN_1000T_FD	(PHY_SPEED_1G | PHY_DUPLEX_FULL)	1000BASE-T 全二重	r_eth_phy.h
PHY_REG_****	---	PHY 標準レジスタアドレス	r_eth_phy.h
PHY_CONTROL_****	---	PHY Mode Control レジスタ bit アサイン	r_eth_phy.h
PHY_STATUS_****	---	PHY Mode Status レジスタ bit アサイン	r_eth_phy.h
PHY_ANADV_****	---	PHY Auto-negotiation advertisement レジスタ bit アサイン	r_eth_phy.h
PHY_ANLINK_****	---	PHY Auto-negotiation link partner ability レジスタ bit アサイン	r_eth_phy.h

定義名	設定値	内容	定義ファイル
ETHSW_LUT_D_PORT0	0	ルックアップテーブル動的レコード ポート 0	r_eth_sw.h
ETHSW_LUT_D_PORT1	1	ルックアップテーブル動的レコード ポート 1	r_eth_sw.h
ETHSW_LUT_D_PORT2	2	ルックアップテーブル動的レコード ポート 2	r_eth_sw.h
ETHSW_LUT_S_PORT0	1	ルックアップテーブル静的レコード ポート 0	r_eth_sw.h
ETHSW_LUT_S_PORT1	2	ルックアップテーブル静的レコード ポート 1	r_eth_sw.h
ETHSW_LUT_S_PORT2	4	ルックアップテーブル静的レコード ポート 2	r_eth_sw.h
LRN_CHK_CNT	10	MAC アドレス学習 IF 連続読みこみ上限	r_eth_sw.h

5.5　コンフィギュレーション

以下のコンフィギュレーションを、用途に合わせて変更してください。

コンフィギュレーション一覧

定義名	内容	定義ファイル
USE_ETHSW	イーサネットスイッチ機能オプション	r_eth.h
USE_ETHSW_MGTAG	イーサネットスイッチ　マネージメント TAG 機能オプション	r_eth.h
MULTICAST_MODE_ENA	マルチキャストフレーム受信オプション	r_eth.h
PROMISCUOUS_MODE_ENA	全てのフレーム受信オプション	r_eth.h
USE_NET_PORT2	使用するイーサネットポート選択オプション	r_eth.h
PHY_ADR0	PHY レジスタアクセス用 PHY0 アドレス	r_eth_phy.h
PHY_ADR1	PHY レジスタアクセス用 PHY1 アドレス	r_eth_phy.h

5.5.1　コンフィギュレーション詳細

(1)　USE_ETHSW

イーサネットスイッチ機能の使用有無を設定します。

・イーサネットスイッチ機能の有効(=1)　：　Default

2 ポート PHY インタフェースが有効となり、外部にスイッチングハブを使用することなく、ライン型、またはリング型のネットワークトポロジを構築することが可能です。評価ボードでは EtherCAT1(J1)が汎用 Ethernet ポート 0、EtherCAT2(J2)が汎用 Ethernet ポート 1 として使用可能となります。

・イーサネットスイッチ機能の無効(=0)

評価ボードでは EtherCAT2(J2)のみが汎用 Ethernet ポートとして使用可能となります。

(2)　USE_ETHSW_MGTAG

イーサネットスイッチ　マネージメント TAG 機能の使用有無を設定します。

本機能を使用する場合はイーサネットスイッチ機能(USE_ETHSW)の有効化が必須です。

・イーサネットスイッチ　マネージメント TAG 機能の有効(=1)　：　Default

マネージメントタグの挿入が許可され、受信フレームが通過した汎用 Ethernet ポートの取得、送信フレームの送信先汎用 Ethernet ポートの指定が可能となります。

・イーサネットスイッチ　マネージメント TAG 機能の無効(=0)

マネージメントタグの挿入がされません。

(3) MULTICAST_MODE_ENA

マルチキャストフレームの受信可否を設定します。

・全てのマルチキャストフレームの受信(=0)

・設定されたマルチキャストフレームのみの受信(=1) : Default

(4) PROMISCUOUS_MODE_ENA

全てのフレームの受信可否を設定します

・全ての Ethernet フレームの受信(=0)

・アドレスフィルタリグされたフレームの受信(= 1) : Default

(5) USE_NET_PORT2

使用する Ethernet ポートを切り替えます。

・汎用 Ethernet ポート 0、1 使用(=0) : Default

・汎用 Ethernet ポート 2 使用(=1)

汎用 Ethernet ポート 0、1 が使用不可となり、評価ボードの EtherMAC(J7)が汎用 Ethernet ポートとして使用可能となります。

PHY アドレスは Ethernet ポート 0 と同一となります。

(6) PHY_ADR

EthernetPHY のアドレスを設定します。

ハードウェア設定でそれぞれの PHY に設定されている PHY アドレスと同一の設定値としてください。

評価ボードでは汎用 Ethernet ポート 0(PHY アドレス= 1)、汎用 Ethernet ポート 1(PHY アドレス=2)となります。

・PHY_ADR0

汎用 Ethernet ポート 0、2 の PHY レジスタアクセス用 PHY アドレス

・PHY_ADR1

汎用 Ethernet ポート 1 の PHY レジスタアクセス用 PHY アドレス

5.6　構造体／共用体一覧

「表 5-4　構造体／共用体一覧」を以下に示します。詳細は「5.6.1 構造体／共用体詳細」で説明します。

表 5-4　構造体／共用体一覧

構造体/共用体型定義	概要	定義ファイル
eth_frm_t 構造体	Ethernet Frame 構造体	r_eth.h
eth_frmvlan_t 構造体	VLAN タグ付き Ethernet Frame 構造体	r_eth.h
ip4_pkt_t 構造体	IPv4 Packet 構造体	r_eth.h
eth_frminfo_t 構造体	Ethernet Frame 情報構造体	r_eth_mac.h
eth_macinfo_t 構造体	MAC Address 情報構造体	r_eth_mac.h
eth_txdesc_t 構造体	送信処理用デスクリプタ構造体	r_eth_mac.h

5.6.1　構造体／共用体詳細

表 5-5　eth_frm_t 構造体

メンバ名	内容
uint8_t dst[6]	Destination MAC Address
uint8_t src[6]	Source MAC Address
uint16_t type	Ethernet Type
uint8_t data[2]	Payload Data 先頭

表 5-6　eth_frmvlan_t 構造体

メンバ名	内容
uint8_t dst[6]	Destination MAC Address
uint8_t src[6]	Source MAC Address
uint32_t vlan	VLAN タグ
uint16_t type	Ethernet Type
uint8_t data[2]	Payload Data 先頭

表 5-7　ip4_pkt_t 構造体

メンバ名	内容
uint8_t ver_ihl	Version & Internet Header Length
uint8_t tos	Type of Service
uint16_t tl	Total Length
uint16_t ident	Identification
uint16_t vcf_fo	Flags & Fragment Offset
uint8_t ttl	Time to Live
uint8_t prtcl	Protocol Type
uint16_t hc	Header Checksum
uint32_t sa	Source IP Address
uint32_t da	Destination IP Address
uint8_t data[2]	Payload 先頭

表 5-8　eth_frminfo_t 構造体

メンバ名			内容
info	uint8_t BYTE		フレーム情報
		uint8_t vlan :1	VLAN タグ有無フラグ(0:VLAN タグ無 , 1:VLAN タグ有)
		uint8_t tcpchk:1	IPv4、TCP、UDP チェックサムのソフトウェアチェックサム計算フラグ 0: HW チェックサム計算結果が確定 (IPv4、TCP、UDP チェックサム値は正しい) 1: HW チェックサム計算結果が未確定 (ソフトウェアによるチェックサム計算が必要) ※受信フレーム時のみ有効
		uint8_t :5	Reserved
uint8_t port			送受信ポート ※イーサネットスイッチ　マネージメント TAG 機能が有効時 受信フレームに対してはフレームが通過した汎用 Ethernet ポート番号 送信フレームに対してはフレームを送信したい汎用 Ethernet ポート番号 が格納されます
uint32_t frm_len			フレーム長
uint8_t *frm			送受信データ格納先バッファへのポインタ

表 5-9　eth_macinfo_t 構造体

メンバ名	内容
uint8_t mac[6]	MAC Address

表 5-10　eth_txdesc_t 構造体

メンバ名	内容
uint32_t addr	転送開始元アドレス
uint32_t len	転送量(バイト)

5.7　エラーコード

ドライバのエラーコードは、0 または正数が正常ケースで負数がエラーとなります。

5.8　関数一覧

「表 5-11　関数一覧」を以下に示します。

表 5-11　関数一覧

関数名	概要	スコープ	定義ファイル
R_ETH_Init	EthernetMAC 初期化	global	r_eth_mac.c
R_ETH_Rcv	Ethernet Frame 送信	global	r_eth_mac.c
R_ETH_Snd	Ethernet Frame 受信	global	r_eth_mac.c
R_ETH_UpdateMode	EthernetMAC 動作モード設定	global	r_eth_mac.c
R_ETH_SetPhyreg	EthernetPHY レジスタ設定	global	r_eth_mac.c
R_ETH_GetPhyreg	EthernetPHY レジスタ取得	global	r_eth_mac.c
R_ETH_memcpy	Ethernet ドライバ用メモリコピー	global	r_eth_mac.c
R_ETH_GetEvent	Ethernet 発生イベント取得	global	r_eth_mac.c
R_ETH_ClrEvent	Ethernet 発生イベントクリア	global	r_eth_mac.c
ntohl	32bit バイトオーダー変換	global	r_eth_mac.c
ntohs	16bit バイトオーダー変換	global	r_eth_mac.c
dmac_memcpy	DMA コピー	local	r_eth_mac.c
eth_wait	指定時間 Wait	local	r_eth_mac.c
eth_int_init	Ethernet 関連割り込み初期化	local	r_eth_mac.c
ETHDMAIR_isr	Ether MACDMA 受信完了割り込みハンドラ	local	r_eth_mac.c
ETHDRIE_isr	Ether MACDMA 受信エラー割り込みハンドラ	local	r_eth_mac.c
ETHRFIV_isr	Ether RX-FIFO オーバーフロー	local	r_eth_mac.c
ETHDMAIT_isr	Ether MACDMA 送信完了割り込みハンドラ	local	r_eth_mac.c
ETHDTIE_isr	Ether MACDMA 送信エラー割り込みハンドラ	local	r_eth_mac.c
ETHIT_isr	Ether 送信完了割り込みハンドラ	local	r_eth_mac.c
ETHTFIU_isr	Ether TX-FIFO アンダーフロー割り込みハンドラ	local	r_eth_mac.c
ETHTFIE_isr	Ether TX-FIFO エラー割り込みハンドラ	local	r_eth_mac.c
R_PHY_Init	EthernetPHY 初期化	global	r_eth_phy.c
R_PHY_Reset	EthernetPHY リセット	global	r_eth_phy.c
R_PHY_SetMode	EthernetPHY 動作モード設定	global	r_eth_phy.c
R_PHY_GetMode	EthernetPHY 動作モード取得	global	r_eth_phy.c
R_PHY_Link	EthernetPHY Link-up	global	r_eth_phy.c
R_ETHSW_Init	EthernetSwitch 初期化	global	r_eth_sw.c
R_ETHSW_AddMaclut	EthernetSwitch アドレステーブル動的レコード追加	global	r_eth_sw.c
R_ETHSW_AddMaclutStatic	EthernetSwitch アドレステーブル静的レコード追加	global	r_eth_sw.c
R_ETHSW_MacLearning	EthernetSwitch MAC アドレス学習	global	r_eth_sw.c
ethsw_update_maclut	EthernetSwitch アドレステーブル更新	local	r_eth_sw.c
ethsw_getCRC8	ハッシュキー取得(CRC8 計算)	local	r_eth_sw.c
ethsw_cap_timer	EthernetSwitch タイマのキャプチャ	local	r_eth_sw.c

5.8.1 EthernetMAC 関数詳細

(1) R_ETH_Init

R_ETH_Init	
概　要	EthernetMAC の初期化を行う
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	int32_t R_ETH_Init(eth_macinfo_t *macinfo)
引　数	eth_macinfo_t *macinfo　：登録 MAC アドレステーブルへのポインタ
戻り値	0　：正常終了 -1　：パラメータエラー
実行条件	● 全てのEthernet系ドライバAPI関数に先だって一度だけ実行
説　明	本関数は下記の処理を行います。 ・登録 MAC アドレステーブルの有効チェック。 ・Ethernet 関連クロック設定、モジュールのストップ解除 ・EthernetMAC 初期化 ・Ethernet 関連モジュールのリセット解除 ・ハードウェアファンクションセットアップ ・EthernetSwitch、EthernetPHY の初期化 ・MAC アドレス設定 ・送信ドライバ用バッファ確保 ・Ethernet 関連割り込みの初期化 ・Ethernet Frame 受信許可
エラー条件	● MACアドレステーブルに定数「MACINFO_UCAST_MAX」を超えるユニキャストMACアドレスが登録されていた場合
補　足	なし

使用例	eth_macinfo_tmacAddr_t[] = {
	{ 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC },　/* Unicast MAC Address */
	{ 0x01, 0x80, 0xC2, 0x00, 0x00, 0x0E },　/* Multicast MAC Address */
	{ MACINFO_TBLEND,MACINFO_TBLEND,MACINFO_TBLEND,
	MACINFO_TBLEND ,MACINFO_TBLEND ,MACINFO_TBLEND }
	};
	
	int32_t errcd;
	
	errcd = R_ETH_Init(macAddr_t);

(2)　R_ETH_Rcv

R_ETH_Rcv		
概　要	Ethernet Frame の受信を行う	
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h	
宣　言	int32_t R_ETH_Rcv(eth_frminfo_t *frminfo)	
引　数	eth_frminfo_t *frminfo	：受信 Ethernet Frame 情報へのポインタ
戻り値	>0	：受信データサイズ
	0	：受信データなし
	-1	：無効フレーム受信
実行条件	● 初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・受信 Buffer 情報レジスタを参照し有効データの受信チェック ・受信フレームのヘッダチェック ・パラメータ「frminfo」が持つバッファへ受信フレームのコピー ・受信フレーム情報の取得 ・受信フレームが格納されていたバッファの解放	
エラー条件	● 無効フレームの受信 ● IP,TCP/UDPヘッダのチェックサム異常 ● 受信フレームのコピーエラー ● 受信フレーム格納先バッファの解放エラー	
補　足	Ethernet Frame 情報のフレーム情報(tcpchk)により IPv4 ヘッダチェックサムと TCP、UDP チェックサムが Ethernet Accelerator のハードウェアチェックサム機能により確定したか確認できます。未確定の場合はソフトウェアによるチェックサム計算が必要です。	

使用例

```
uint8_t          rcv_buf[1514];
eth_frminfo_t    rxfrm = {0};
int32_t          errcd;

rxfrm.frm = rcv_buf;

errcd = R_ETH_Rcv(&rxfrm);
if(0 < errcd){
     /*  受信データ有り  */
} else if( 0 > errcd){
     /*  無効フレーム受信  */
}
```

(3) R_ETH_Snd

R_ETH_Snd	
概　要	Ethernet Frame の送信を行う
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	int32_t R_ETH_Snd(eth_frminfo_t *frminfo)
引　数	eth_frminfo_t *frminfo　：送信 Ethernet Frame 情報へのポインタ
戻り値	0　：送信完了 -1　：送信エラー
実行条件	● 初期化関数R_ETH_Init()実行後
説　明	本関数は下記の処理を行います。 ・パラメータ「frminfo」が持つバッファから送信バッファへ送信フレームのコピー ・送信フレームコントローラ情報の設定 ・送信フレームデスクリプタの設定 ・送信用 DMA 起動 ・Ethernet Frame 送信完了待ち ・エラーステータス取得
エラー条件	● バッファ間コピーエラー ● 送信用DMA起動エラー ● 送信エラー
補　足	Ethernet Accelerator 機能が有効な場合、フラグメントパケットではないパケットの IPv4 ヘッダチェックサム、TCP/UDP チェックサムは Ethernet Accelerator のハードウェアチェックサム機能により、送信時に自動計算されます。

使用例	uint8_t　snd_buf[1514];
	eth_frminfo_t　txfrm = {0};
	int32_t　errcd;
	eth_frm_t　*ethfrm_tx;
	
	txfrm.frm = snd_buf;
	
	ethfrm_tx = (eth_frm_t *) txfrm.frm;
	
	R_ETH_memcpy(ethfrm_tx->src,/* Source MAC Addr */,6);
	R_ETH_memcpy(ethfrm_tx->dst,/* Destination MAC Addr */,6);
	ethfrm_tx->type = htons(/* Ethernet Type */);
	R_ETH_memcpy(ethfrm_tx->data,/* Payload Data */,/* Payload Data Length */);
	#if(USE_ETHSW & USE_ETHSW_MGTAG)
	/* Select the transfer port */
	txfrm .port = ETH_PORT***;
	#endif
	txfrm .frm_len = /* Ethernet Frame Length */
	errcd = R_ETH_Snd(&txfrm);
	if(0 > errcd){
	　/* 送信エラー */
	} else {
	　/* 送信完了 */
	}

(4)　R_ETH_UpdateMode

R_ETH_UpdateMode	
概　要	EthernetMAC の動作モードを設定する
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	int32_t R_ETH_UpdateMode(uint8_t port)
引　数	uint8_t port　：動作モード設定先ポート ：　ETH_PORT0 ：　ETH_PORT1
戻り値	0　：送信完了 -1　：送信エラー
実行条件	●　初期化関数R_ETH_Init()実行後
説　明	本関数は下記の処理を行います。 ・引数「port」にて指定の EthernetPHY が Link-up 状態であれば、PHY の設定状態に合わせて EthernetMAC の動作モード設定を行う
エラー条件	●　引数「port」が範囲外
補　足	なし

使用例	int32_t　errcd;
	
	errcd = R_ETH_UpdateMode(ETH_PORT0);
	if(0 > errcd){
	　/* 設定エラー */
	} else {
	　/* 設定完了 */
	}

(5)　R_ETH_SetPhyreg

R_ETH_SetPhyreg	
概　要	EthernetPHY レジスタへの設定を行う
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void R_ETH_SetPhyreg(uint8_t phyadr, uint8_t regadr, uint16_t val)
引　数	uint8_t phyadr　：PHY アドレス uint8_t regadr　：設定先 PHY レジスタアドレス uint16_t val　：PHY レジスタへの設定値
戻り値	なし
実行条件	●　初期化関数R_ETH_Init()実行後
説　明	本関数は下記の処理を行います。 ・引数「phyadr」にて指定された PHY のレジスタ「regadr」に対して「val」を設定する
エラー条件	なし
補　足	なし
使用例	R_ETH_SetPhyreg(PHY_ADR0, PHY_REG_CONTROL, PHY_CONTROL_SPEED_100M);

(6) R_ETH_GetPhyreg

R_ETH_GetPhyreg		
概　要	EthernetPHY レジスタの取得を行う	
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h	
宣　言	uint16_t R_ETH_GetPhyreg(uint8_t phyadr, uint8_t regadr)	
引　数	uint8_t phyadr	: PHY アドレス
	uint8_t regadr	: 取得先 PHY レジスタアドレス
戻り値	PHY レジスタの参照結果	
実行条件	● 初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・引数「phyadr」にて指定された PHY のレジスタ「regadr」を読み込み、結果を返す	
エラー条件	なし	
補　足	なし	
使用例	uint16_t data; data = R_ETH_SetPhyreg(PHY_ADR0,PHY_REG_CONTROL);	

(7) R_ETH_memcpy

R_ETH_memcpy		
概　要	Ethernet 系ドライバ用メモリコピー	
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h	
宣　言	int32_t R_ETH_memcpy(void *dst, const void *src, uint32_t n)	
引　数	void *dst	: コピー先アドレス
	const void *src	: コピー元アドレス
	uint32_t n	: コピーサイズ
戻り値	0	: 正常終了
実行条件	なし	
説　明	本関数は下記の処理を行います。 ・DMA コピー関数を呼び出す。	
エラー条件	なし	
補　足	なし	
使用例	R_ETH_memcpy(/* Destination Addr */ , /* Source Addr */ , /* Copy Length */);	

(8)　R_ETH_GetEvent

R_ETH_GetEvent		
概　要	Ethernet 系発生イベント取得	
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h	
宣　言	uint32_t R_ETH_GetEvent(void)	
引　数	なし	
戻り値	正常ケース	: bit22 : 1= ETH_EVT_TXFIFOERR : bit21 : 1= ETH_EVT_TXFIFOUDF : bit20 : 1= ETH_EVT_TXCMP : bit17 : 1= ETH_EVT_TXDMAERR : bit16 : 1= ETH_EVT_TXDMACMP : bit10 : 1= ETH_EVT_RXFIFOOVF : bit9 : 1= ETH_EVT_RXDMAERR : bit8 : 1= ETH_EVT_RXDMACMP
実行条件	なし	
説　明	本関数は下記の処理を行います。 ・Ethernet 系モジュールにより発生したイベントを取得する。	
エラー条件	なし	
補　足	なし	
使用例	uint32_t EventFlag; EventFlag = R_ETH_GetEvent();	

(9)　R_ETH_ClrEvent

R_ETH_ClrEvent		
概　要	Ethernet 系発生イベントクリア	
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h	
宣　言	void R_ETH_ClrEvent(uint32_t clrflg)	
引　数	uint32_t clrflg	: クリア可能イベント : ETH_EVT_RXDMACMP : ETH_EVT_RXDMAERR : ETH_EVT_RXFIFOOVF : ETH_EVT_TXDMACMP : ETH_EVT_TXDMAERR : ETH_EVT_TXCMP : ETH_EVT_TXFIFOUDF : ETH_EVT_TXFIFOERR
戻り値	なし	
実行条件	なし	
説　明	本関数は下記の処理を行います。 ・Ethernet 系モジュールにより発生したイベントのクリアを行う。	
エラー条件	なし	
補　足	なし	
使用例	R_ETH_ClrEvent(ETH_EVT_TXDMACMP | ETH_EVT_TXCMP);	

(10) ntohl

ntohl	
概　要	32bit のバイトオーダー変換
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	uint32_t ntohl(uint32_t val)
引　数	uint32_t val　　：バイトオーダー変換値
戻り値	バイトオーダー変換結果
実行条件	なし
説　明	本関数は下記の処理を行います。 ・32bit ネットワークバイトオーダーをホストバイトオーダーに変換する。
エラー条件	なし
補　足	なし
使用例	uint32_t data; data = ntohl(0x11223344);

(11) ntohs

ntohs	
概　要	16bit のバイトオーダー変換
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	uint16_t ntohl(uint16_t val)
引　数	uint16_t val　　：バイトオーダー変換値
戻り値	バイトオーダー変換結果
実行条件	なし
説　明	本関数は下記の処理を行います。 ・16bit ネットワークバイトオーダーをホストバイトオーダーに変換する。
エラー条件	なし
補　足	なし
使用例	uint16_t data; data = ntohs(0x1122);

(12) dmac_memcpy

dmac_memcpy	
概　要	DMA Copy
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	static void *dmac_memcpy(void *dst, const void *src, uint32_t n)
引　数	void *dst　：コピー先アドレス const void *src　：コピー元アドレス uint32_t n　：コピーサイズ
戻り値	コピー先アドレス
実行条件	なし
説　明	本関数は下記の処理を行います。 ・DMAC レジスタを設定し、DMA 転送を行う。
エラー条件	なし
補　足	なし
使用例	なし

(13) eth_wait

eth_wait	
概　要	一定時間 Wait
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	static void eth_wait(uint32_t value)
引　数	uint32_t value　：Wait 時間(us)
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・指定時間から大よその処理サイクル数を算出し、ループ処理を行いタイマ不使用の Wait 処理を実現する。
エラー条件	なし
補　足	なし
使用例	なし

(14) eth_int_init

eth_int_init	
概　要	Ethernet 関連割り込み初期化
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	static void eth_int_init(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ethernet 関連割り込みハンドラの登録を行う。 下記割り込みハンドラを登録する。 Ether MACDMA 受信完了割り込みハンドラ Ether MACDMA 受信エラー割り込みハンドラ Ether RX-FIFO オーバーフロー割り込みハンドラ Ether MACDMA 送信完了割り込みハンドラ Ether MACDMA 送信エラー割り込みハンドラ Ether 送信完了割り込みハンドラ Ether TX-FIFO アンダーフロー割り込みハンドラ Ether TX-FIFO エラー割り込みハンドラ
エラー条件	なし
補　足	なし
使用例	なし

(15) ETHDMAIR_isr

ETHDMAIR_isr	
概　要	Ether MACDMA 受信完了割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHDMAIR_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether MACDMA 受信完了割り込みハンドラ Ether MACDMA 受信完了イベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(16) ETHDRIE_isr

ETHDRIE_isr	
概　要	Ether MACDMA 受信エラー割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHDRIE_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether MACDMA 受信エラー割り込みハンドラ 　Ether MACDMA 受信エラーイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(17) ETHRFIV_isr

ETHRFIV_isr	
概　要	Ether RX-FIFO オーバーフロー割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHRFIV_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether RX-FIFO オーバーフロー割り込みハンドラ 　Ether RX-FIFO オーバーフローイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(18) ETHDMAIT_isr

ETHDMAIT_isr	
概　要	Ether MACDMA 送信完了割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHDMAIT_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether MACDMA 送信完了割り込みハンドラ 　Ether MACDMA 送信完了イベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(19) ETHDTIE_isr

ETHDTIE_isr	
概　要	Ether MACDMA 送信エラー割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHDTIE_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether MACDMA 送信エラー割り込みハンドラ 　Ether MACDMA 送信エラーイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(20) ETHIT_isr

ETHIT_isr	
概　要	Ether 送信完了割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHIT_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether 送信完了割り込みハンドラ 　Ether 送信完了割り込みイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(21) ETHTFIU_isr

ETHTFIU_isr	
概　要	Ether TX-FIFO アンダーフロー割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHTFIU_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether TX-FIFO アンダーフロー割り込みハンドラ 　Ether TX-FIFO アンダーフローイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

(22) ETHTFIE_isr

ETHTFIE_isr	
概　要	Ether TX-FIFO エラー割り込みハンドラ
ヘッダ	r_eth.h r_eth_sw.h eth_hwfnc.h
宣　言	void ETHTFIE_isr(void)
引　数	なし
戻り値	なし
実行条件	なし
説　明	本関数は下記の処理を行います。 ・Ether TX-FIFO エラー割り込みハンドラ 　Ether TX-FIFO エラー割り込みイベントの設定を行う。
エラー条件	なし
補　足	なし
使用例	なし

5.8.2 EthernetPHY 関数詳細

(1) R_PHY_Init

R_PHY_Init		
概　要	EthernetPHY の初期化を行う	
ヘッダ	r_eth.h	
宣　言	int32_t R_PHY_Init(uint8_t phyadr)	
引　数	uint8_t phyadr	: 初期化先 PHY
		: PHY_ADR0
		: PHY_ADR1
戻り値	0	: 正常終了
	-1	: パラメータエラー
実行条件	● 初期化関数R_ETH_Init()実行時に呼ばれる	
説　明	本関数は下記の処理を行います。	
	・EthernetPHY リセット	
	・EthernetPHY 初期設定	
エラー条件	● 引数「phyadr」が範囲外	
補　足	なし	
使用例	int32_t errcd;	
	errcd = R_PHY_Init(PHY_ADR0);	

(2) R_PHY_Reset

R_PHY_Reset		
概　要	EthernetPHY のリセットを行う	
ヘッダ	r_eth.h	
宣　言	int32_t R_PHY_Reset(uint8_t phyadr)	
引　数	uint8_t phyadr	: リセット先 PHY アドレス
		: PHY_ADR0
		: PHY_ADR1
戻り値	0	: 正常終了
	-1	: パラメータエラー
実行条件	なし	
説　明	本関数は下記の処理を行います。	
	・リセット先 PHY のリセットを行う。	
エラー条件	● 引数「phyadr」が範囲外	
補　足	なし	
使用例	int32_t errcd;	
	errcd = R_PHY_Reset(PHY_ADR0);	

(3) R_PHY_SetMode

R_PHY_SetMode		
概　要	EthernetPHY の動作モード設定を行う	
ヘッダ	r_eth.h	
宣　言	int32_t R_PHY_SetMode(uint8_t phyadr, uint16_t mode , uint8_t nego)	
引　数	uint8_t phyadr	: 動作モード設定先 PHY アドレス
		: PHY_ADR0
		: PHY_ADR1
	uint16_t mode	: 動作モード
		: LAN_10T_HD
		: LAN_10T_FD
		: LAN_100TX_HD
		: LAN_100TX_FD
	uint8_t nego	: Auto-Negotiation Mode
		: 0 = Disable Auto-Negotiation
		: 1 = Enable Auto-Negotiation
戻り値	0	: 正常終了
	-1	: パラメータエラー
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・PHY アドレス、Auto-Negotiation Mode に従って動作モードを設定する。	
エラー条件	● 引数「phyadr」が範囲外	
補　足	PHY レジスタへのアクセスは EthernetMAC の R_ETH_SetPhyreg() ドライバを使用	
使用例	int32_t errcd; errcd = R_PHY_SetMode(PHY_ADR0, LAN_100TX_HD,0);	

(4) R_PHY_GetMode

R_PHY_GetMode		
概　要	EthernetPHY の動作モード取得を行う	
ヘッダ	r_eth.h	
宣　言	int32_t R_PHY_GetMode(uint8_t phyadr)	
引　数	uint8_t phyadr	: 動作モード取得先 PHY アドレス : PHY_ADR0 : PHY_ADR1
戻り値	正常ケース	: bit31 : 0 = No Error : bit14 : Link Status : 0 = Link-Down : 1 = Link-Up : bit13 : Auto-Negotiation Status : 0 = Auto-Negotiation Disable : 1 = Auto-Negotiation Enable : bit2-1 : Speed : 01b = 10BASE-T : 10b = 100BASE-T : bit0 : Duplex : 0 = Half Duplex : 1 = Full Duplex
	-1	: パラメータエラー
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・PHY アドレスを元に、Link 状態、Auto-Negotiation 設定状態を PHY レジスタから取得する	
エラー条件	● 引数「phyadr」が範囲外	
補　足	PHY レジスタへのアクセスは EhternetMAC の R_ETH_SetPhyreg() ドライバを使用	

使用例	int32_t status;
	
	status = R_PHY_GetMode(PHY_ADR0);
	if((status & PHY_LINK_MASK) == PHY_LINK_UP){
	　/* PHY Link-up */
	}else{
	　/* PHY Link-down */
	}	

(5)　R_PHY_Link

R_PHY_Link		
概　要	EthernetPHY の Link-up 確認と動作モード更新	
ヘッダ	r_eth.h	
宣　言	int32_t R_PHY_Link(uint8_t phyadr)	
引　数	uint8_t phyadr	: Link-up 確認先 PHY アドレス : PHY_ADR0 : PHY_ADR1
戻り値	正常ケース	: bit31　:　0 = No Error :　bit14　: Link Status :　　0 = Link-Down :　　1 = Link-Up :　bit13　: Auto-Negotiation Status :　　0 = Auto-Negotiation Disable :　　1 = Auto-Negotiation Enable :　bit2-1　: Speed :　　01b = 10BASE-T :　　10b = 100BASE-T :　bit0　: Duplex :　　0 = Half Duplex :　　1 = Full Duplex
	-1	: パラメータエラー
実行条件	●　EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・PHY アドレスを元に、EthernetPHY の動作モードを取得 ・前回取得時とモードが違う、かつ Link-up 状態の場合は EthernetMAC の R_ETH_UpdateMode() ドライバを呼び出し EthernetMAC の動作モードを設定する ・EthernetPHY の動作モードを返す	
エラー条件	●　引数「phyadr」が範囲外	
補　足	PHY レジスタへのアクセスは EhternetMAC の R_ETH_SetPhyreg() ドライバを使用	

使用例	int32_t status;
	uint8_t　　rcv_buf[1514];
	eth_frminfo_t　rxfrm = {0};
	int32_t　　errcd;
	
	rxfrm.frm = rcv_buf;
	
	status = R_PHY_Link(PHY_ADR0);
	if((status & PHY_LINK_MASK) == PHY_LINK_UP){
	　　errcd = R_ETH_Rcv(&rxfrm);
	}	

5.8.3 EthernetSwitch 関数詳細

(1) R_ETHSW_Init

R_ETHSW_Init	
概　要	EthernetSwitch の初期化を行う
ヘッダ	r_eth.h r_eth_sw.h
宣　言	int32_t R_ETHSW_Init(eth_macinfo_t *macinfo , uint8_t hub)
引　数	eth_macinfo_t *macinfo　：登録 MAC アドレステーブルへのポインタ uint8_t hub　：スイッチングハブ機能 0 = スイッチングハブ機能無効 1 = スイッチングハブ機能有効
戻り値	0　：正常終了
実行条件	● 初期化関数R_ETH_Init()実行時に呼ばれる
説　明	本関数は下記の処理を行います。 ・EthernetSwitch の各機能(MAC,スイッチ,タイマなど)の初期設定を行う。 ・引数「hub」によりハブ機能の設定を行う。
エラー条件	なし
補　足	なし

使用例	eth_macinfo_tmacAddr_t[] = {
	{ 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC }, /* Unicast MAC Address */
	{ 0x01, 0x80, 0xC2, 0x00, 0x00, 0x0E }, /* Multicast MAC Address */
	{ MACINFO_TBLEND,MACINFO_TBLEND,MACINFO_TBLEND,
	MACINFO_TBLEND ,MACINFO_TBLEND ,MACINFO_TBLEND }
	};
	
	int32_t errcd;
	
	errcd = R_ETHSW_Init(macAddr_t,1);

(2)　R_ETHSW_AddMaclut

R_ETHSW_AddMaclut		
概　要	EthernetSwitch アドレステーブルへの動的 MAC レコードの追加	
ヘッダ	r_eth.h r_eth_sw.h	
宣　言	int32_t R_ETHSW_AddMaclut(uint8_t *mac , uint8_t port)	
引　数	uint8_t *mac	：MAC アドレスへのポインタ
	uint8_t port	：EtherSW Dynamic port No. ETHSW_LUT_D_PORT0 = Port 0 ETHSW_LUT_D_PORT1 = Port 1 ETHSW_LUT_D_PORT2 = Port 2(CPU Port) ※複合指定不可
戻り値	>=0	:アドレステーブルへのエントリ位置(ハッシュキー)
	-1	：MAC レコード登録エラー
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・引数「mac」、「port」から動的 MAC レコードを作成。 ・MAC アドレスからハッシュキーを取得しアドレステーブルへ追加。	
エラー条件	アドレステーブルに空きがなく MAC アドレスが登録不可の場合	
補　足	なし	
使用例	uint8_t mac [] = { 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC }; int32_t hash; hash = R_ETHSW_AddMaclut(mac,ETHSW_LUT_D_PORT2);	

(3) R_ETHSW_AddMaclutStatic

R_ETHSW_AddMaclutStatic		
概　要	EthernetSwitch アドレステーブルへの静的 MAC レコードの追加	
ヘッダ	r_eth.h r_eth_sw.h	
宣　言	int32_t R_ETHSW_AddMaclutStatic(uint8_t *mac , uint8_t port)	
引　数	uint8_t *mac	: MAC アドレスへのポインタ
	uint8_t port	: EtherSW Static port No. ETHSW_LUT_S_PORT0 = ポート 0 ETHSW_LUT_S_PORT1 = ポート 1 ETHSW_LUT_S_PORT2 = ポート 2 ※複合指定可能
戻り値	>=0	: アドレステーブルへのエントリ位置(ハッシュキー)
	-1	: MAC レコード登録エラー
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・引数「mac」、「port」から静的 MAC レコードを作成。 ・MAC アドレスからハッシュキーを取得しアドレステーブルへ追加。	
エラー条件	アドレステーブルに空きがなく MAC アドレスが登録不可の場合	
補　足	なし	
使用例	uint8_t mac [] = { 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC }; int32_t hash; hash = R_ETHSW_AddMaclutStatic (mac,ETHSW_LUT_S_PORT2);	

(4) R_ETHSW_MacLearning

R_ETHSW_MacLearning		
概　要	EthernetSwitch MAC アドレス学習	
ヘッダ	r_eth.h r_eth_sw.h	
宣　言	int32_t R_ETHSW_MacLearning(void)	
引　数	なし	
戻り値	0	: MAC アドレス学習有り
	-1	: MAC アドレス学習無し
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後	
説　明	本関数は下記の処理を行います。 ・ラーニングインターフェースが Ethernet ポートを通過したフレームの MAC アドレスを学習していれば MAC アドレス、フレームが通過した Ethernet ポート、ハッシュキーを取得 ・動的 MAC レコードを作成し、アドレステーブルへ追加する	
エラー条件	なし	
補　足	なし	
使用例	R_ETHSW_MacLearning();	

(5) ethsw_update_maclut

ethsw_update_maclut	
概　要	EthernetSwitch アドレステーブル更新
ヘッダ	r_eth.h r_eth_sw.h
宣　言	static int32_t ethsw_update_maclut(RIN_ETHSW_LUT_ADR_Typedef *macinfo,uint8_t hash)
引　数	RIN_ETHSW_LUT_ADR_Typedef *macinfo　：MAC レコードへのポインタ uint8_t hash　：アドレステーブルへの追加位置(ハッシュキー)
戻り値	>=0　：アドレステーブルへの追加位置(ハッシュキー) -1　：MAC レコード登録エラー
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後
説　明	本関数は下記の処理を行います。 ・アドレステーブルで、引数「hash」位置から 8 エントリ以内に空きエントリがにあれば MAC レコードを追加 ・アドレステーブルで、引数「hash」位置から 8 エントリ以内に同一 MAC アドレスのエントリがあれば MAC レコードを上書き ・同一 MAC アドレスまたは空きのエントリがない場合はタイムスタンプが一番古いエントリへ MAC レコードを上書き
エラー条件	8 エントリ全て静的 MAC レコードで埋まっており、動的 MAC レコードが追加できない場合
補　足	なし
使用例	なし

(6) ethsw_getCRC8

ethsw_getCRC8	
概　要	EthernetSwitch CRC8 によるハッシュキー算出
ヘッダ	r_eth.h r_eth_sw.h
宣　言	static crc ethsw_getCRC8(uint8_t *buff , uint16_t size)
引　数	uint8_t *buff　：ハッシュキー算出元データへのポインタ uint16_t size　：ハッシュキー算出元データサイズ
戻り値	uint8_t　：ハッシュキー
実行条件	なし
説　明	本関数は下記の処理を行います。 ・引数「buff」から CRC8 によるハッシュキーを算出する
エラー条件	なし
補　足	なし
使用例	なし

(7) ethsw_cap_timer

ethsw_cap_timer	
概　要	EthernetSwitch タイマーのキャプチャ
ヘッダ	r_eth.h r_eth_sw.h
宣　言	static void ethsw_cap_timer(void)
引　数	なし
戻り値	なし
実行条件	● EthernetMAC初期化関数R_ETH_Init()実行後
説　明	本関数は下記の処理を行います。 ・EthernetSwitch が持つタイマ値のキャプチャ指示を行う ・キャプチャ指示によりタイマレジスタが最新に更新される
エラー条件	なし
補　足	なし
使用例	なし

5.9 EthernetMAC 初期化手順

EthernetMAC のレジスタ初期設定から受信許可となるまでの手順を以下に示します。

5.9.1 EthernetMAC 初期化フロー

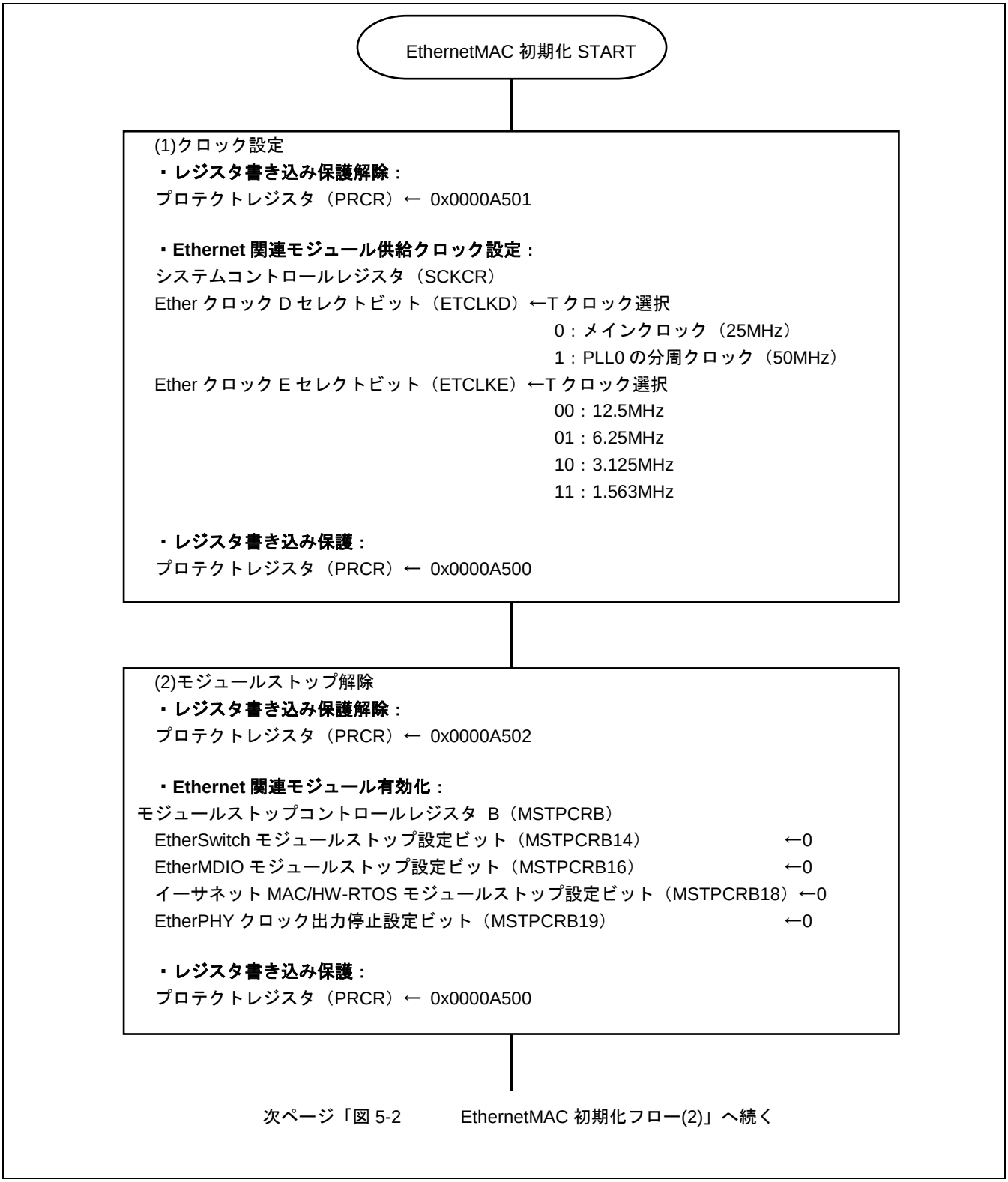


図 5-1　EthernetMAC 初期化フロー(1)

前ページ「図 5-1　　EthernetMAC 初期化フロー(1)」より

(3)EthernetMAC 動作モード設定レジスタプロテクト解除
・レジスタ書き込み保護解除：
イーサネットシステムプロテクトコマンドレジスタ（ETSPCMD）← 0x000000A5
イーサネットシステムプロテクトコマンドレジスタ（ETSPCMD）← 0x00000001
イーサネットシステムプロテクトコマンドレジスタ（ETSPCMD）← 0x0000FFFE
イーサネットシステムプロテクトコマンドレジスタ（ETSPCMD）← 0x00000001

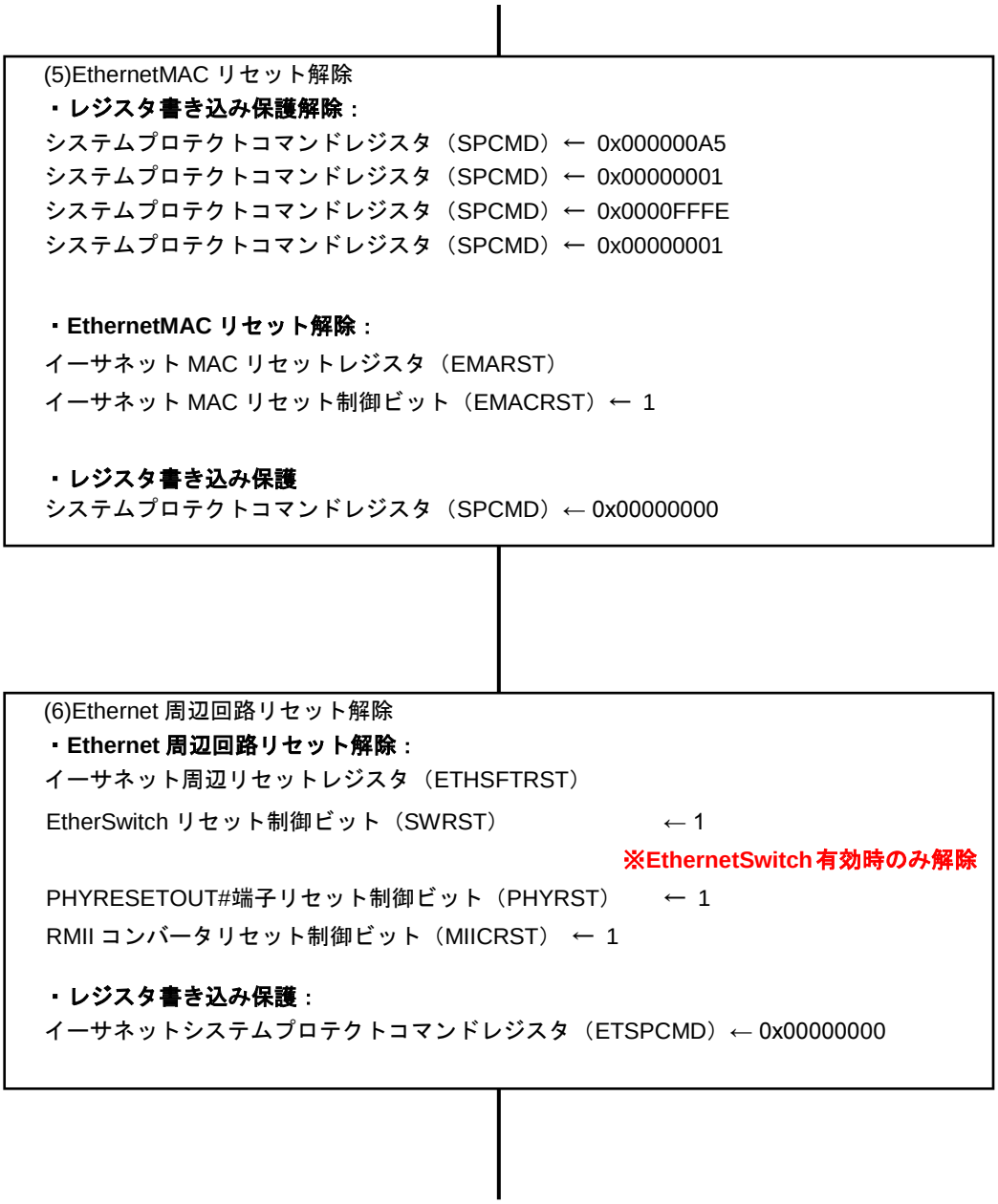
(4)イーサネットインタフェース設定
・イーサネットインタフェース機能選択：
MAC セレクトレジスタ（MACSEL）
イーサネット MAC モード選択ビット（MAC）← インタフェース選択
000：Media I/F Port0／Port1 有効
（EthernetSwitch 機能有効）
011：Media IF Port1 のみ有効
（EthernetSwitch 機能無効）

・MII ／ RMII コンバータの制御設定：
MII コントロールレジスタ（MII_CTRLn）（n = 0 ～ 1）
MII コンバータモード選択ビット（MODE）　← モード設定
x0000：MII モード
10100：RMII モード（10Mbps）
10101：RMII モード（100Mbps）
DuplexMode 設定ビット（FULLD）　← モード設定
0：half duplex
1：full duplex
CRS 判定条件選択ビット（RMII_CRS_MODE）← 条件設定
0：「CRS | TXEN」
1：「CRS | RXDV | TXEN」
・イーサネット PHY LINK モード設定：
イーサネット PHY LINK モードレジスタ（ETHPHYLNK）
EtherSwitch 使用時の PHYLINK0 端子のアクティブレベル切り替えビット（SWLINK0）← 1
EtherSwitch 使用時の PHYLINK1 端子のアクティブレベル切り替えビット（SWLINK1）← 1

次ページ「図 5-3　　EthernetMAC 初期化フロー(3)」へ続く

図 5-2　　EthernetMAC 初期化フロー(2)

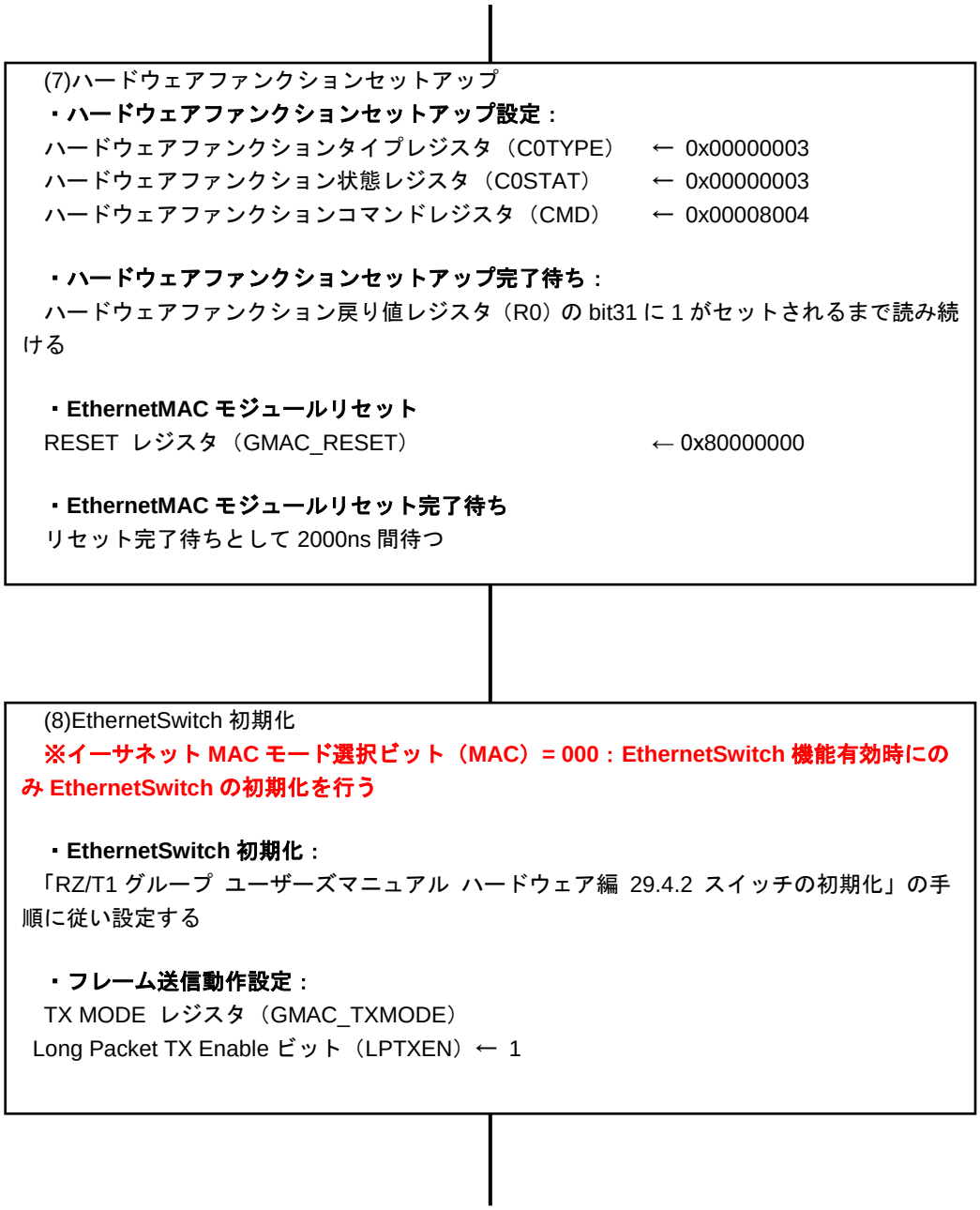
前ページ「図 5-2　EthernetMAC 初期化フロー(2)」より



次ページ「図 5-4　EthernetMAC 初期化フロー(4)」へ続く

図 5-3　EthernetMAC 初期化フロー(3)

前ページ「図 5-3　　EthernetMAC 初期化フロー(3)」より



次ページ「図 5-5　　EthernetMAC 初期化フロー(5)」へ続く

図 5-4　　EthernetMAC 初期化フロー(4)

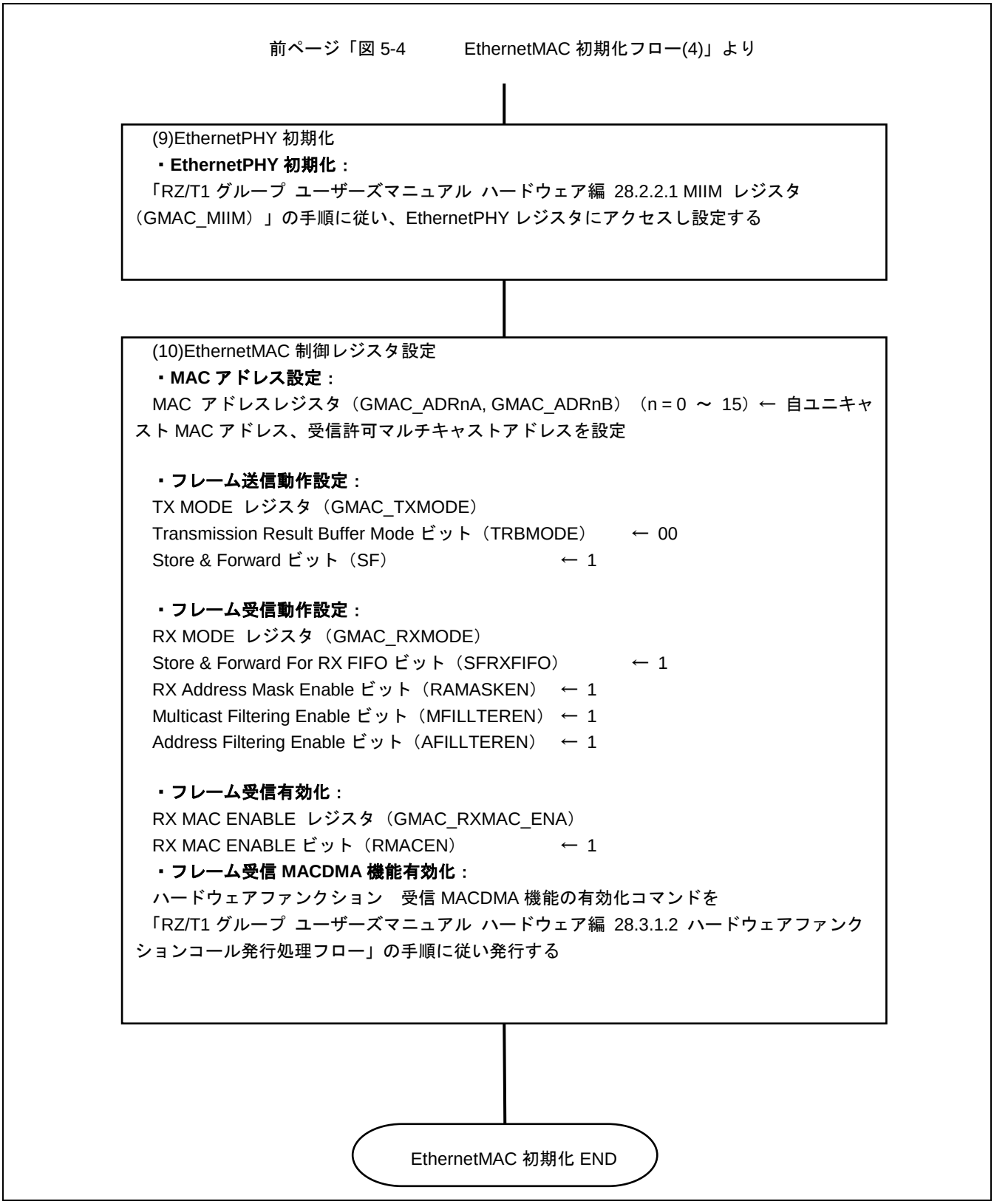


図 5-5　EthernetMAC 初期化フロー(5)

6. ソフトウェア説明(アプリケーション)

6.1 アプリケーション概要

本サンプルプログラムでは、Ethenet 通信制御ドライバを用いて、Ethernet フレームの送受信やフレームの簡易解析による ping 応答機能を提供します。

6.2 ソフトウェア・ブロック構成

「図 6-1　Ethernet アプリケーション・ソフトウェア・ブロック図」を以下に示します。

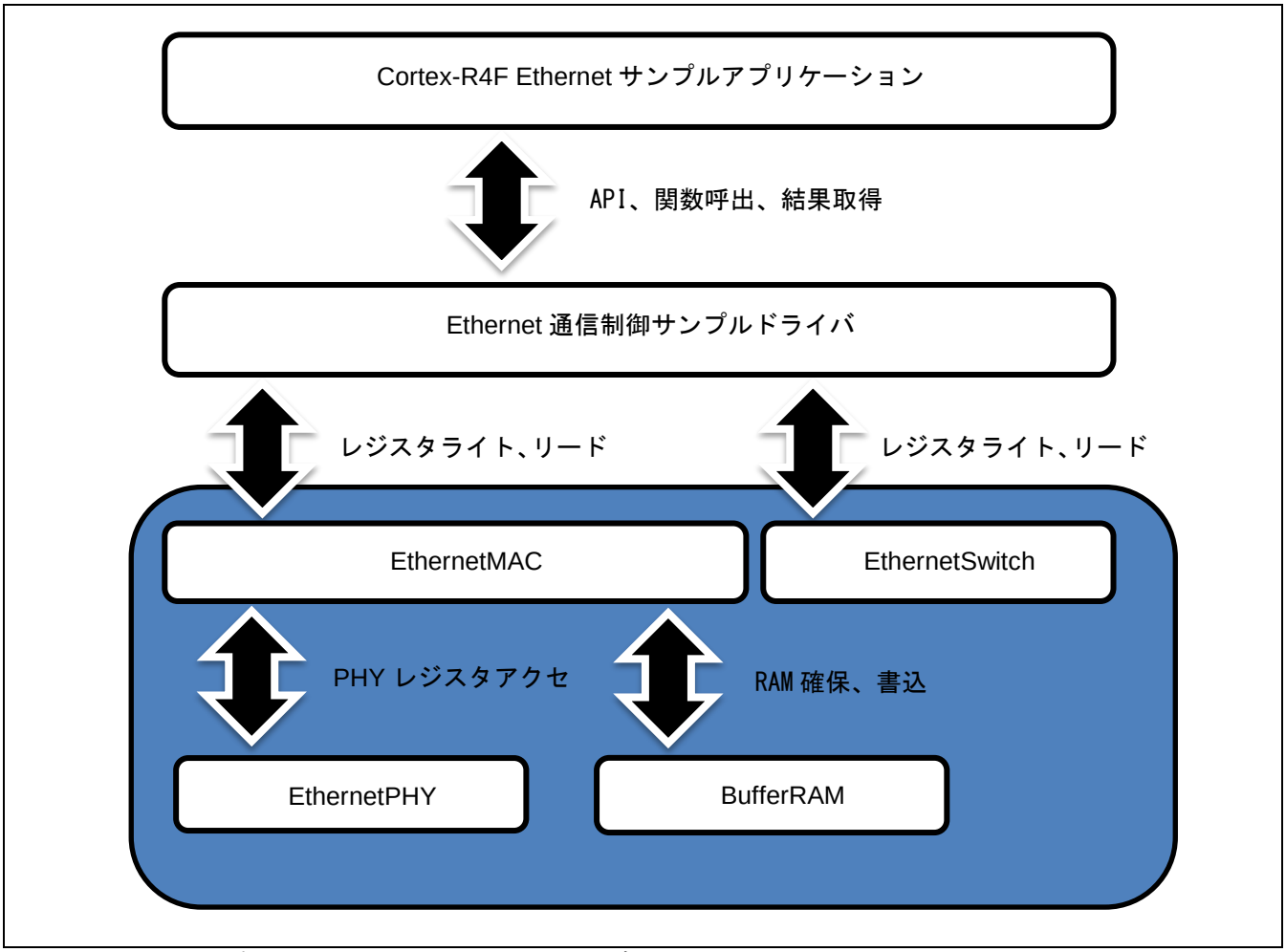


図 6-1 Ethernet アプリケーション・ソフトウェア・ブロック図

6.3 ファイル一覧

サンプルプログラムで使用しているアプリケーションを以下に示します。

表 6-1　サンプルプログラムで使用するファイル

ファイル名	概要	備考
.. *1/src/sample/main.c	Ethernet サンプルプログラムメイン処理	
.. *1/src/sample/app.c	Etnernet サンプルアプリケーション	

【注】　..*1　使用するサンプルプログラムのフォルダ名です。

6.4　定数一覧

「表 6-2　定数一覧」を以下に示します。

表 6-2　定数一覧

定義名	設定値	内容	定義ファイル
ARP_HRD_ETH	0x0001	ARP Hardware Type : Ethernet	app.c
ARP_PRO_IP	0x0800	ARP Protocol Type : IP	app.c
ARP_OP_REQ	0x0001	ARP Operation : Request	app.c
ARP_OP_REP	0x0002	ARP Operation : Reply	app.c
ARP_PKT_SIZE	28	ARP Packet Size	app.c
IP_PRO_ICMP	1	IP Protocol Type : ICMP	app.c
IP_PRO_TCP	6	IP Protocol Type : TCP	app.c
IP_PRO_UDP	17	IP Protocol Type : UDP	app.c
IP_HEAD_SIZE	20	IP Packet Header Size	app.c
ICMP_TYPE_ECHO_REP	0	ICMP Type : Echo Reply	app.c
ICMP_TYPE_ECHO_REQ	8	ICMP Type : Echo Request	app.c

6.5　構造体／共用体一覧

「表 6-3　構造体／共用体一覧」を以下に示します。詳細は「6.5.1 構造体／共用体詳細」で説明します。

表 6-3　構造体／共用体一覧

構造体/共用体型定義	概要	定義ファイル
arp_pkt_t 構造体	ARP Packet 構造体	app.c
icmp_echo_pkt_t 構造体	ICMP Packet 構造体	app.c

6.5.1　構造体／共用体詳細

表 6-4　arp_pkt_t 構造体

メンバ名	内容
uint16_t hwtype	Hardware Type
uint16_t ptype	Protocol Type
uint8_t hwlen	Hardware Address Length
uint8_t plen	Protocol Address Length
uint16_t op	Operation Code
uint8_t sha[6]	Source Hardware Address(MAC Address)
uint8_t spa[4]	Source Protocol Address(IP Address)
uint8_t dha[6]	Destination Hardware Address(MAC Address)
uint8_t dpa[4]	Destination Protocol Address(IP Address)

表 6-5　icmp_echo_pkt_t 構造体

メンバ名	内容
uint8_t type	ICMP Type
uint8_t code	Type of Service
uint16_t cs	Total Length
uint16_t ident	Identifier
uint16_T seq	Sequence Number
uint8_t data[2]	Payload 先頭

6.6　大域変数一覧

「表 6-6　変数一覧」を以下に示します。

表 6-6　変数一覧

型	変数名	概要	定義ファイル
eth_macinfo_t	macAddr_t[]	MAC アドレス情報テーブル	main.c
uint8_t	rcv_buf[1514]	Ethernet Frame 受信バッファ	main.c
uint8_t	snd_buf[1514]	Ethernet Frame 送信バッファ	main.c
eth_frm_t *	ethfrm_rx	受信 Ethernet Frame 参照用ポインタ	app.c
eth_frm_t *	ethfrm_tx	送信 Ethernet Frame 参照用ポインタ	app.c
uint32_t	lcl_ipa	ローカル IP アドレス	app.c

6.7　エラーコード

アプリケーションのエラーコードは、0 または正数が正常ケースで負数がエラーとなります。

6.8　関数一覧

「表 6-7　関数一覧」を以下に示します。

表 6-7　関数一覧

関数名	概要	スコープ	定義ファイル
main	メイン処理	global	main.c
app_main	Ethernet Frame 受信アプリケーションメイン	local	app.c
app_arp_recv	ARP Packet 受信アプリケーション	local	app.c
app_ip4_recv	IPv4 Packet 受信アプリケーション	local	app.c
app_icmp_recv	ICMP Packet 受信アプリケーション	local	app.c
app_ip4_echo_res	IPv4 Packet Echo Response アプリケーション	local	app.c
app_echo_res	Echo Response アプリケーション	local	app.c
checksum	チェックサム算出	local	app.c

6.8.1　関数詳細

(1)　main

main	
概　要	メイン処理
ヘッダ	r_eth.h r_eth_sw.h
宣　言	int main (void)
引　数	なし
戻り値	なし
実行条件	●　ブート処理後に実行される
説　明	本関数は下記の処理を行います。 ・サンプルプログラム共通初期化 ・Ethernet 関連モジュール初期化 ・送受信バッファ情報設定 ・EtherentPHY Link-up チェック ・EthernetSwitch MAC アドレス学習 ・Ethernet 関連イベント取得 ・EthernetFrame 受信チェック ・受信 EthenerFrame 解析
エラー条件	なし
補　足	なし
使用例	なし

(2) app_main

app_main		
概　要	Ethernet Frame 受信アプリケーションメイン処理	
ヘッダ	r_eth.h	
宣　言	int32_t app_main(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● Ethernet Frame受信時にメイン処理から実行	
説　明	本関数は下記の処理を行います。 ・受信 Ethernet Frame のフレーム解析を行う ・Ethernet Type が IPv4 の場合は IPv4 Packet 受信アプリケーションを呼び出す ・Ethernet Type が ARP の場合は ARP Packet 受信アプリケーションを呼び出す ・Ethernet Type がどちらでもない場合は Echo Response アプリケーションを呼び出す。	
エラー条件	なし	
補　足	なし	
使用例	なし	

(3) app_arp_recv

app_arp_recv		
概　要	ARP Packet 受信アプリケーション	
ヘッダ	r_eth.h	
宣　言	static int32_t app_arp_recv(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● ARPパケット受信時にEthernet Frame受信アプリケーションメイン処理から実行	
説　明	本関数は下記の処理を行います。 ・ARP(Address Resolution Protocol)パケットのヘッダ解析を行う ・ハードウェアタイプ、プロトコルタイプ、IP アドレスが一致した場合に応答パケットを作成し、返送する ・ハードウェアタイプ、プロトコルタイプ、IP アドレスのいずれかが一致しない場合は何も行わない。	
エラー条件	API エラー発生時	
補　足	なし	
使用例	なし	

(4) app_ip4_recv

app_ip4_recv		
概　要	IPv4 Packet 受信アプリケーション	
ヘッダ	r_eth.h	
宣　言	static int32_t app_ip4_recv(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● IPv4パケット受信時にEthernet Frame受信アプリケーションメイン処理から実行	
説　明	本関数は下記の処理を行います。 ・IPv4 パケットのヘッダ解析を行う ・プロトコルタイプが ICMP の場合は ICMP Packet 受信アプリケーションを呼び出す。 ・プロトコルタイプが TCP、UDP の場合は IPv4 Echo Response アプリケーションを呼び出す。 ・プロトコルタイプが ICMP、TCP、UDP 以外の場合は Echo Response アプリケーションを呼び出す。	
エラー条件	なし	
補　足	なし	
使用例	なし	

(5) app_icmp_recv

app_icmp_recv		
概　要	ICMP Packet 受信アプリケーション	
ヘッダ	r_eth.h	
宣　言	static int32_t app_icmp_recv(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● ICMPパケット受信時にIPv4 Packet受信アプリケーションから実行	
説　明	本関数は下記の処理を行います。 ・ICMP(Internet Control Message Protocol)パケットのチェックサムチェックを行う。 ・チェックサム異常がなく、ICMP タイプが Echo Request の場合に応答パケットを作成し、返送する。 ・ICMP タイプが Echo Request 以外の場合は何も行わない。	
エラー条件	チェックサム異常、API エラー発生時	
補　足	なし	
使用例	なし	

(6) app_ip4_echo_res

app_ip4_echo_res		
概　要	IPv4 Packet Echo Response アプリケーション	
ヘッダ	r_eth.h	
宣　言	static int32_t app_ip4_echo_res(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● TCP、UDPパケット受信時にIPv4 Packet受信アプリケーションから実行	
説　明	本関数は下記の処理を行います。 ・送信先、送信元 IP アドレス、送信パケットサイズを設定する。 ・受信 IPv4 パケットの Payload 部を送信元へ返送します。	
エラー条件	API エラー発生時	
補　足	なし	
使用例	なし	

(7) app_echo_res

app_echo_res		
概　要	Echo Response アプリケーション	
ヘッダ	r_eth.h	
宣　言	static int32_t app_echo_res(eth_frminfo_t *rx , eth_frminfo_t *tx)	
引　数	eth_frminfo_t *rx	: 受信フレーム情報へのポインタ
	eth_frminfo_t *tx	: 送信フレーム情報へのポインタ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	● サポート外のEthernet Frameを受信した場合	
説　明	本関数は下記の処理を行います。 ・受信フレームの Payload 部を送信元へ返送します。	
エラー条件	API エラー発生時	
補　足	なし	
使用例	なし	

(8) checksum

checksum		
概　要	チェックサム算出	
ヘッダ	r_eth.h	
宣　言	static uint16_t checksum(uint8_t *buf , uint16_t count)	
引　数	uint8_t *buf	: チェックサム計算元データへのポインタ
	eth_frminfo_t *tx	: チェックサム計算元データのサイズ
戻り値	0	: 正常終了
	-1	: API エラー
実行条件	なし	
説　明	本関数は下記の処理を行います。 ・指定データ範囲のチェックサムを算出する。	
エラー条件	なし	
補　足	RFC1071(Internet Checksum)のアルゴリズムを元にしています。	
使用例	なし	

6.9　フローチャート

アプリケーションのフローチャートを示します。

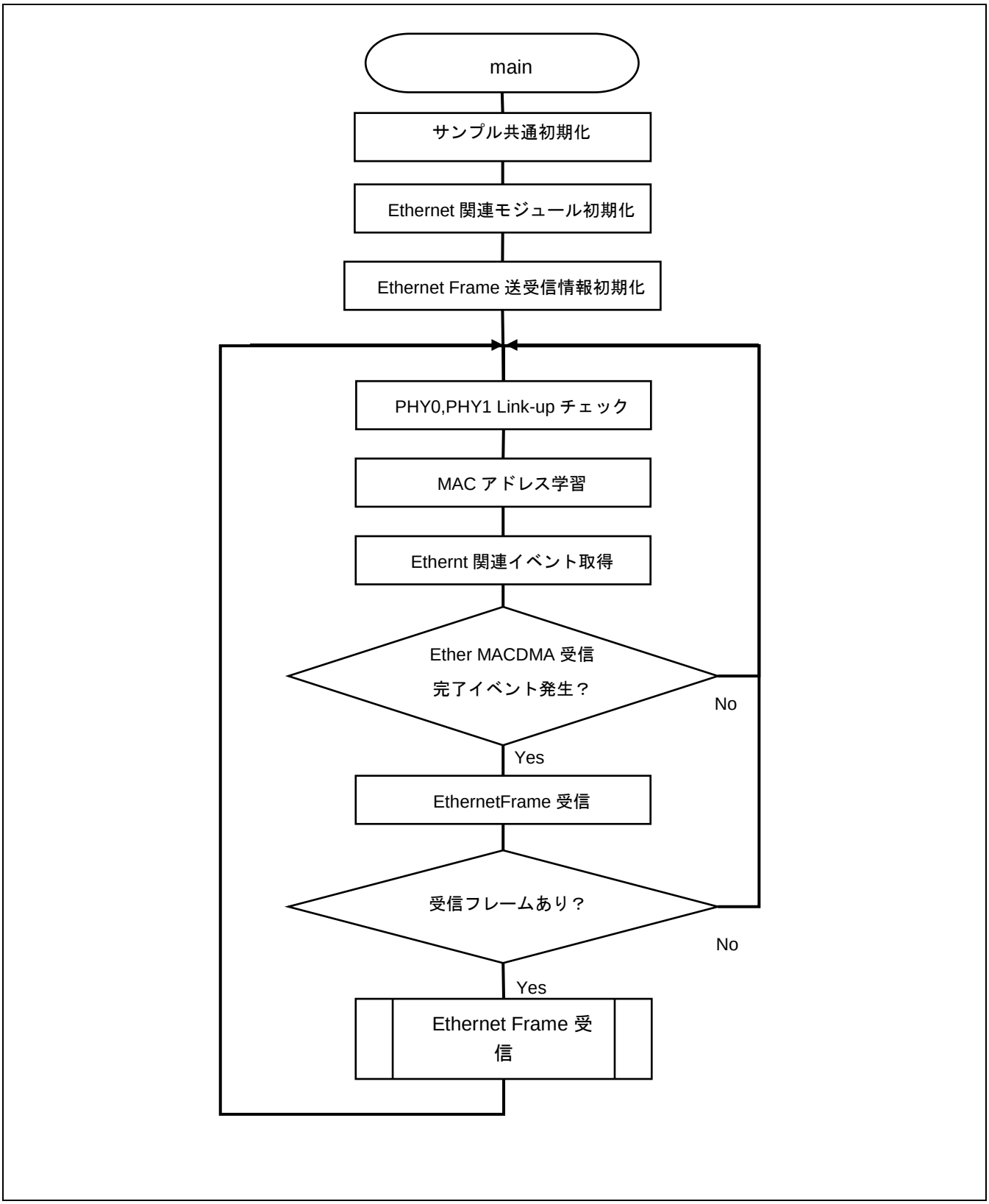


図 6-2 メイン処理(main)

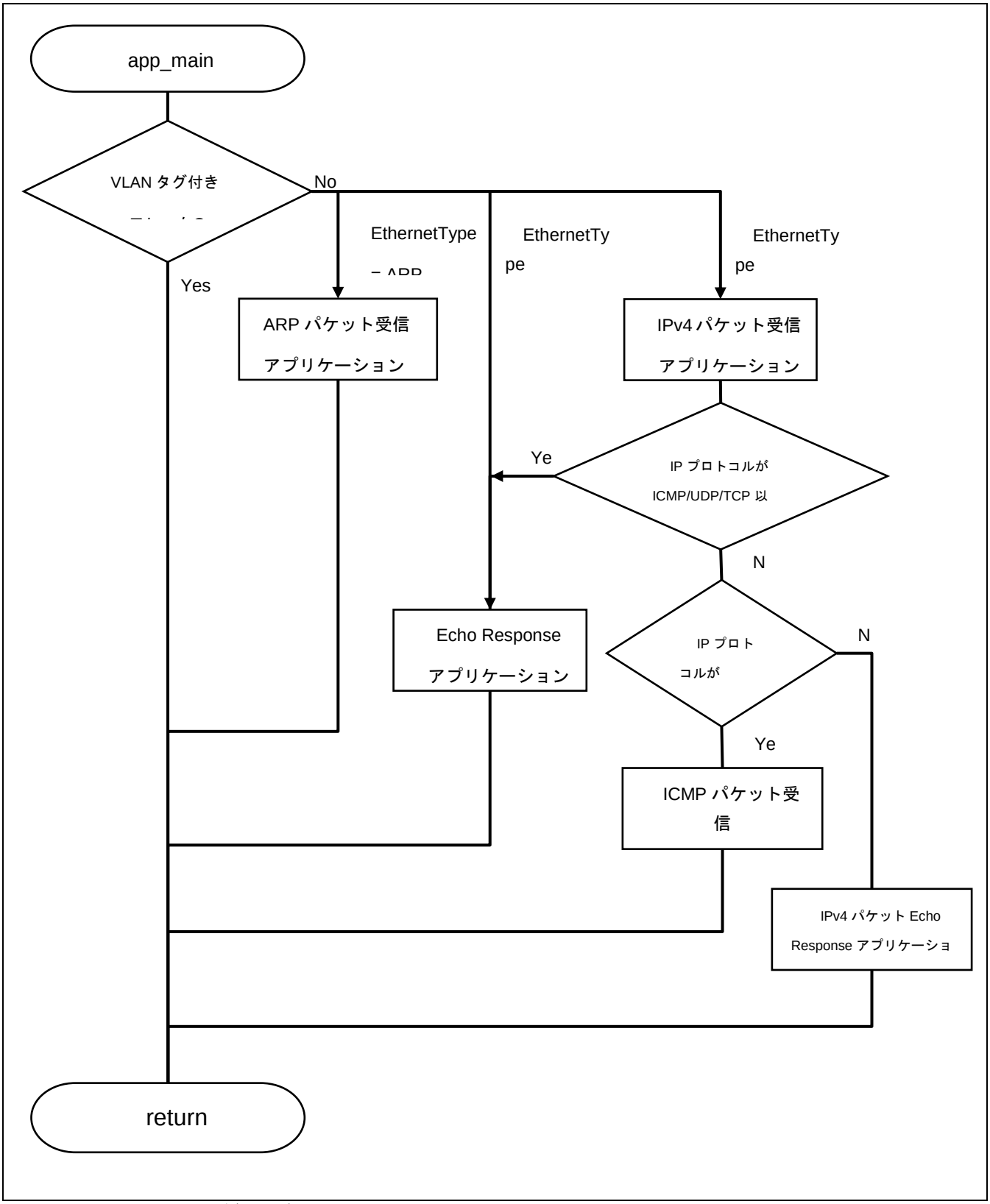


図 6-3 Ethernet Frame 受信アプリケーションメイン(app_main)

7. サンプルプログラム

サンプルプログラムは、ルネサス エレクトロニクスホームページから入手してください。

8. ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://www.renesas.com/

お問合せ先
http://www.renesas.com/contact/

すべての商標および登録商標は，それぞれの所有者に帰属します。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.10.07	-	初版発行
1.10	2017.05.16	10	表 5-3　定数一覧　イベントフラグ、Ethernet Type 追加
		13	5.5　コンフィグレーション　イーサネットポート選択オプション説明追加
		15	5.6　構造体／共用体一覧　IPv4 Packet 構造体追加
		17	5.8　関数一覧　イベントフラグ関連関数追加
		46	6.　ソフトウェア説明(アプリケーション)　IPv4 Packet Echo Response アプリケーション関連追加

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

〇ARM, AMBA, ARM Cortex, ThumbおよびARM Cortex-R4FはARM LimitedのEUおよびその他の国における商標および登録商標です。
〇Ethernetおよびイーサネットは、 富士ゼロックス株式会社の登録商標です。
〇IEEEは、the Institute of Electrical and Electronics Engineers, Inc. の登録商標です。
〇EtherCAT®は、ドイツBeckhoff Automation GmbHによりライセンスされた特許取得済み技術であり商標登録です。
〇TwinCATは、Beckhoff Automation GmbH, Germanyの登録商標です。
〇その他、本資料中の製品名やサービス名は全てそれぞれの所有者にする商標または登録商標です。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、その他の不適切に使用しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、
   金融端末基幹システム、各種安全制御装置等

   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することはできません。たとえ、意図しない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を、(1)核兵器、化学兵器、生物兵器等の大量破壊兵器およびこれらを運搬することができるミサイル（無人航空機を含みます。）の開発、設計、製造、使用もしくは貯蔵等の目的、(2)通常兵器の開発、設計、製造または使用の目的、または(3)その他の国際的な平和および安全の維持の妨げとなる目的で、自ら使用せず、かつ、第三者に使用、販売、譲渡、輸出、賃貸もしくは使用許諾しないでください。

   当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様の転売、貸与等により、本書（本ご注意書きを含みます。）記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は一切その責任を負わず、お客様にかかる使用に基づく当社への請求につき当社を免責いただきます。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載された情報または当社製品に関し、ご不明点がある場合には、当社営業にお問い合わせください。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

(Rev.3.0-1 2016.11)

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社　〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：https://www.renesas.com/contact/

© 2017 Renesas Electronics Corporation. All rights reserved.
Colophon 5.0